

HPCC Workshop

Rui Wang

hpcc@usc.edu

ITS/HPCC, Spring 2011

Topics

- Overview of HPCCC resources
- HPCCC accounts illustrated
- HPCCC job queues
- HPCCC job submission
- HPCCC job monitoring
- Tips of job control
- HPCCC documentation project

HPCC Resources

- Nodes/Processors/Memory
- File System: Disk/Tape – how to access
- Network
- Software Stack

Nodes

- `pbsnodes`: list nodes and their state
- `pbsnodes [nodename|:property ...]`
- Examples:
 - `pbsnodes -a`
 - `pbsnodes hpc1812`
 - `pbsnodes :x86_64`
 - `pbsnodes :x86_64 | grep properties | wc -l`

Nodes cont'd

- Node property
 - Model:sc1425, V20z, pe1950, x2200, X4100, X4600, dx340, dx360, sl160
 - CPU type: opteron, P4
 - # of cores: dualcore, quadcore, hexcore
 - Architecture: x86_64
 - Disk: disk30g, disk60g, disk140g, disk200g

Nodes cont'd

- Total number of nodes: 2775
- All 64 bit nodes
- dualcore nodes: 785
- quadcore nodes: 1462
- hexcore nodes: 528

Nodes cont'd

```
-bash-3.00$ pbsnodes hpc2523
```

```
hpc2523
```

```
state = job-exclusive
```

```
np = 8
```

```
properties = x2200,opteron,quadcore,x86_64,disk60g,m10g,main,large,quick
```

```
ntype = cluster
```

```
jobs = 0/5537095.hpc-pbs.usc.edu, 1/5537095.hpc-pbs.usc.edu, 2/5537095.hpc-  
pbs.usc.edu, 3/5537095.hpc-pbs.usc.edu, 4/5537095.hpc-pbs.usc.edu,  
5/5537095.hpc-pbs.usc.edu, 6/5537095.hpc-pbs.usc.edu, 7/5537095.hpc-  
pbs.usc.edu
```

```
status = arch=x86_64,opsys=linux,uname=Linux hpc2523 2.6.9-78.0.13.ELsmp #1  
SMP Wed Jan 14 15:55:36 EST 2009
```

```
x86_64,sessions=30673,nsessions=1,nusers=1,idletime=5725951,totmem=1747067  
2kb,availmem=13256932kb,physmem=16418456kb,ncpus=8,loadave=0.15,netload=  
4982844832860,size=232417340kb:232511800kb,state=free,jobs=5537095.hpc-  
pbs.usc.edu,rectime=1238444544
```

Nodes cont'd

- Cluster Login Nodes
 - hpc-login1 32 bit
 - hpc-login2 64 bit
- Admin nodes:
 - hpc-pbs resource management
 - hpc-admin lower level infrastructure server
 - hpc-fs4, hpc-fs5, and almaak file server
 - hpc-dns dns server

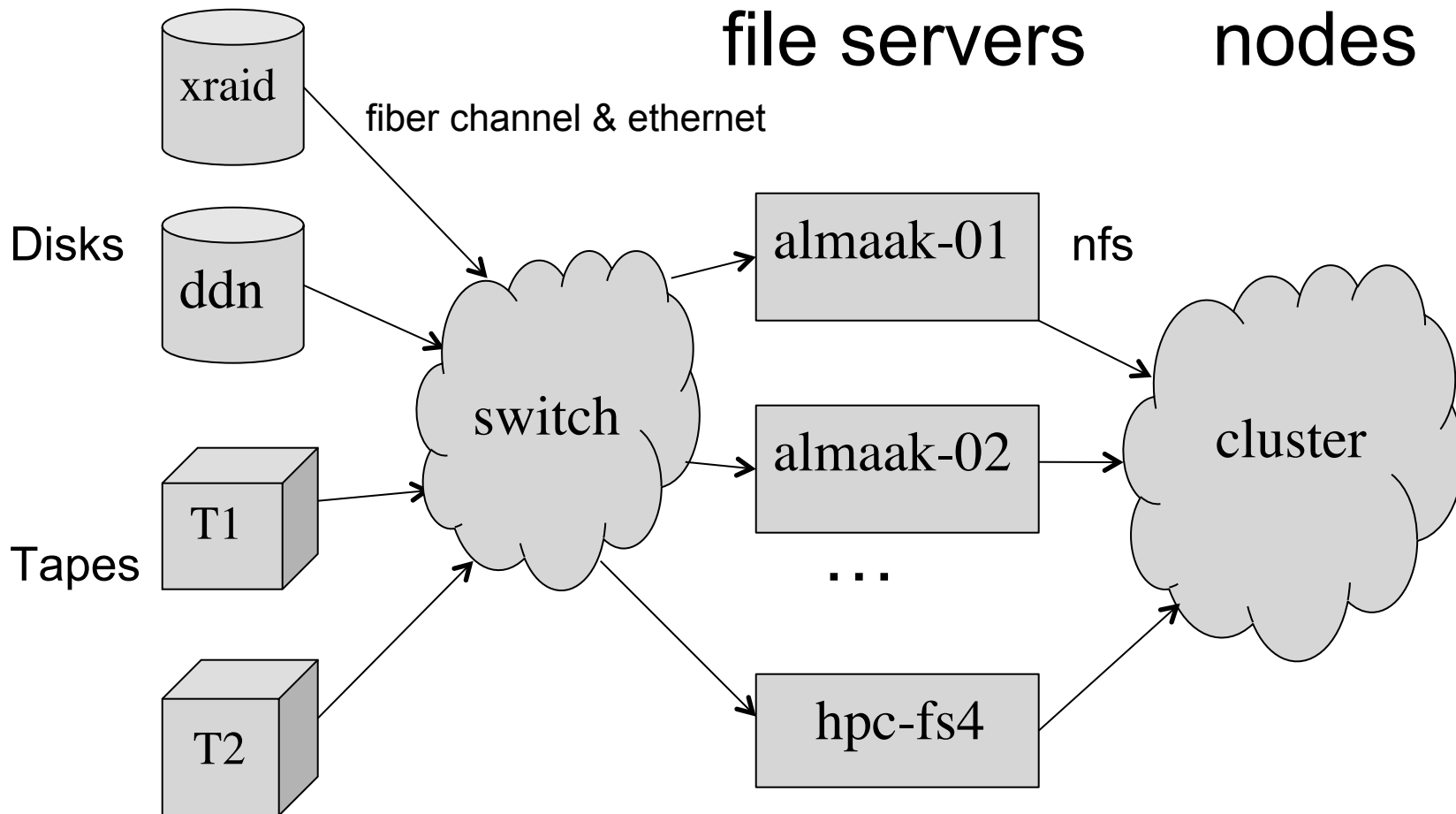
Nodes cont'd

- Cluster Compute Nodes
 - 32 bit job and 64 bit job
- Large memory nodes
 - 22 total
 - 5 in the main queue, each has 64GB ram and 16 dualcore 2.6GHz opteron processors

File System

- File Servers
- Disks
- SAMFS & NFS
- Tapes
- /scratch file system (multi node job)
- home dir & project dir, quotas

File System cont'd



File System cont'd

- Disks: 328TB total, different file systems
- Direct connected to file servers by fiber
- Nodes mount directories from file servers over NFS
- Backup and restore: samfs advantage
 - 19 tape drives

File System cont'd

- Home dir & project dir
 - Research computing facility directory
 - Project has option to put user's home dir on their own disk
 - HPCC Project directory
 - Shared project space, HPCC owns
 - /home/rcf-proj3
 - 1 Terabyte

File System cont'd

– mount path

- For home dir: /home/rcf-xx/userid
- For project dir: /home/rcf-proj3/allocationid/
userid
- Always use /home/rcf-proj3/...
- Avoid use
 - /auto/rcf-proj3/...
 - /export/samfs-proj3/...

File System cont'd

- Quotas
 - Individual quota: 1GB on RCF
 - Directory quota: admin id
- Command showquota
 - Log in `almaak.usc.edu`(file server)
 - `showquota [-a adminid] [-f filesystem]`

File System cont'd

- Examples:

- show individual disk usage

- `showquota -u warshel -f /export/samfs-rcf`

Type	ID	OUse	OSoft	OHard
Files	user	5404	546	50000 51000
Blocks	user	5404	35.48M	50.00M 51.00M

- Show directory disk usage

- `showquota -a 134 -f /home/rcf-proj3`

Type	ID	OUse	OSoft	OHard
Files	admin	134	541	50000 51000
Blocks	admin	134	566.66M	5.00G 5.00G

File System cont'd

- /scratch file system
 - Only multi-node jobs
 - Joining all /tmp on the nodes
 - Destroyed once job exits – remember to copy your files!

File System cont'd

- How to upload/download data to/from hpcc?
 - scp (encryption overhead!)
 - sftp
 - Do it from/to the file server: almaak
 - Path to the project directory is identical on almaak and cluster
 - `scp mydata.tar userid@almaak.usc.edu:/home/rcf-proj3/an/userid/`

File System Cont'd

- Common mistakes
 - Avoid doing research in your home dir
 - Home dirs are for personal use only
 - Most cause for 'quota exceeded' error message
 - Copying files slow
 - Are you doing it on hpc-login nodes?
 - Do it from file server
 - Fiber channel directly connected – fast performance

Network

- Myrinet fabric: 2G & 10G
 - Jobs can not span over
- Cross sectional bandwidth: 70%-80%
- Nodes can not access outside network
 - exceptions can be made (we don't recommend it)

Software stack

- Installed software: /usr/usc
- Software on demand: license needed?
- Install software in your own dir
- X forwarding for applications

Installed software

- should be either already in your \$PATH, or
- check in /usr/usc
 - for all software relatively big
 - for software that doesn't package well – just 1 copy
 - How to use?
 - directory naming convention
 - /usr/usc/software-name/version
 - Alias default
 - run setup script
 - for both bash and c shell
 - set up all necessary environment variables

Installed Software cont'd

- Special case I: Gaussian
 - Permission denied after set up
 - needs approval from PI
- Special case II: mpich
 - Default:mx-gnu
 - Other builds:mx-intel, mx-pgi
 - Compatibility issue

Software on demand

- We can install software on user demand
 - What's the catch?
 - It needs to be reasonably popular
 - We only make sure it runs correctly
 - You need to know how it runs!
 - Does it need a license?
 - Your responsibility to get it
 - Examples: amber, fdtd, etc

Install software on your own

- How to compile a program on linux?
 - Get source code tar ball
 - Uncompress...not in your home dir!
 - `tar zxf mycode.tar.gz`
 - Readme or INSTALL
 - Generate Makefile: run configure script
 - `cd /path/to/program`
 - `./configure`

Install software on your own cont'd

- Compile the source code
 - make
- Make install: need to be careful
 - Default installation directory
 - How to modify parameters
- Postprocessing
 - Add to your \$PATH
- Benefits to do it on your own

Install software on your own cont'd

- LD_LIBRARY_PATH
 - REALLY BAD!!!
 - Cause a lot of load on nfs servers with extra 'stats' call
 - Do NOT set it!
 - How to avoid? Add runtime library search path
 - -Wl,-rpath,-Wl,/home/rcf-proj3/...

Install software on your own cont'd

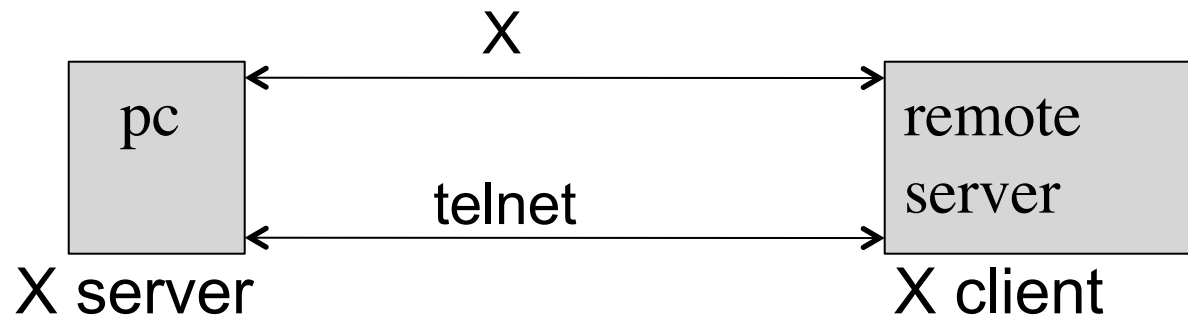
- Command `chrpath`
 - To view the runtime library search path of an executable
 - To change the runtime library search path of an executable...with restrictions!
 - If there is no runtime search path, you can't add one. The updated path can't be longer than previous one

X forwarding for applications

- How do I display remote GUI locally?
 - Make sure a X server is running on your local computer
 - Open a terminal under X environment
 - Use X forwarding provided by ssh
 - `ssh -X remote-computer`
 - Run the application on remote-computer and GUI should appear

Traditional way of X

- Typical senario



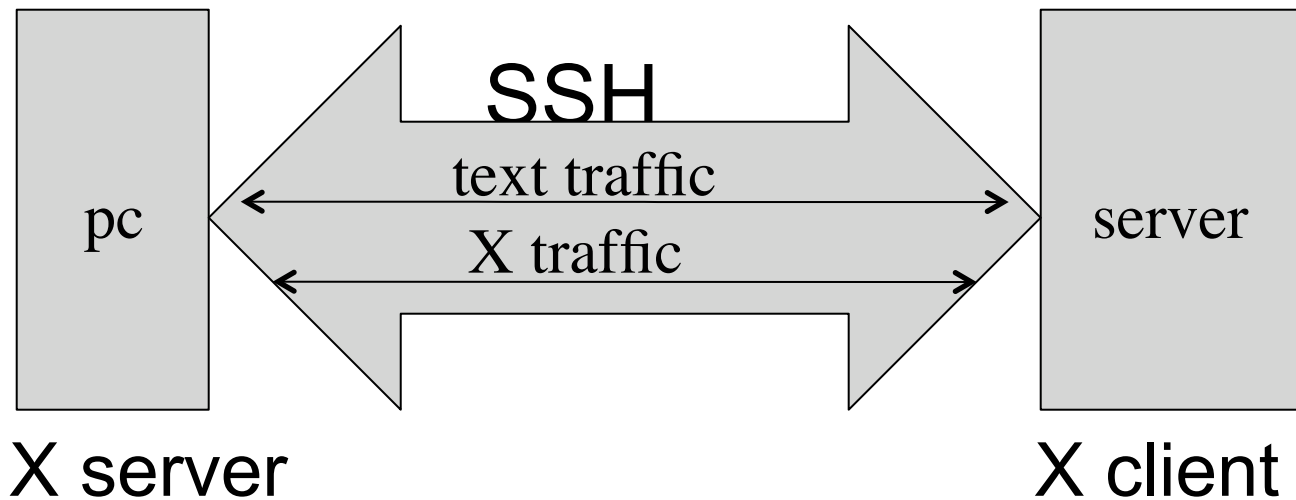
xhost +servername

export DISPLAY=pc:0.0

- Very bad - no authentication

X forwarding on ssh

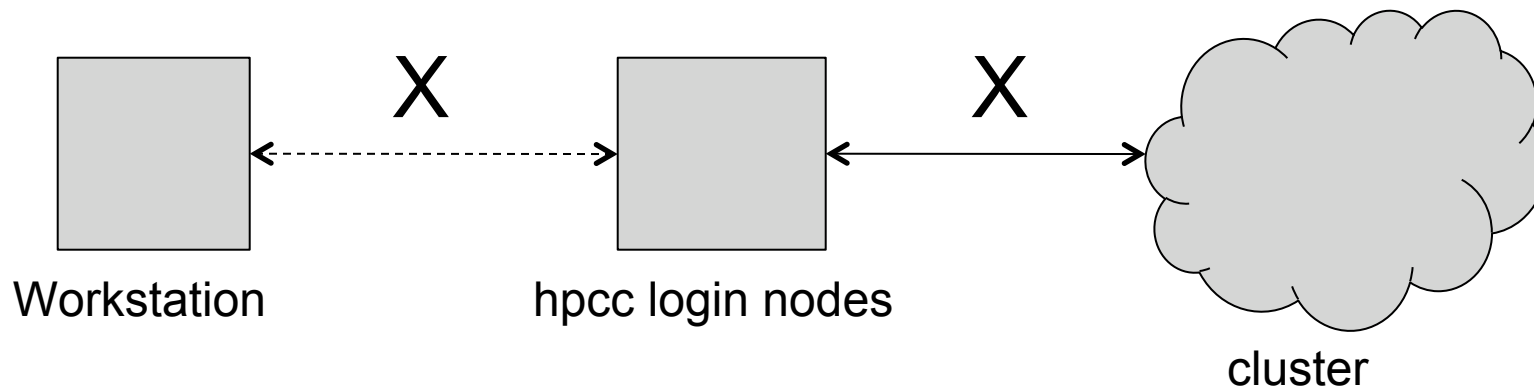
- Typical senario



- It can keep going multi-hops

More on X forwarding

- X forwarding can be activated between compute nodes and head nodes



- User needs to set up rest of the route

More on X forwarding

- Example:
 - open a x terminal on your local workstation
 - `ssh -X userid@hpc-login1.usc.edu`
 - run `xterm`
- Troubleshooting
 - Did you set `$DISPLAY` ? wrong if you set it
 - Is the X server running ?
 - Test: run `xterm` on your own computer

Example: use matlab on hpcc

- Set up X forwarding
 - In your x terminal, type
 - ssh -X userid@hpc-login1.usc.edu
- Test X forwarding
 - After logging in hpc-login1, type
 - xterm
 - xterm window should appear

Example: use matlab on hpcc Cont'd

- Set up matlab
 - In the terminal, type

```
source /usr/usc/matlab/default/setup.sh
```
 - Now, type

```
which matlab
```
 - Output should be

```
/usr/usc/matlab/2006a/bin/matlab
```
 - Matlab is in your \$PATH now

Example: use matlab on hpcc Cont'd

- Now, type
 - matlab
- Matlab window should appear on your desktop

HPCC Account Application

- HPCC webpage
 - <http://www.usc.edu/hpcc>
 - Upcoming events will be posted
- Account applications
 - <https://www-rcf.usc.edu/rcfdocs/hpcc/allocations>
 - Security certificate
 - Log in with your email id & password
 - Fill the form: what are the resources?
 - Once submitted, takes up to one week to process

Prerequisites

- Qbank System
 - group account <lc_xx>, individual account<pechpras>
 - Group account to hold computing time
 - Individual account as member of group account
 - Individual account can be in multiple group accounts
- Research Computing Facility(RCF): almaak
- Unix group affiliation
 - Restriction of login to RCF machines: <hpcusers>
 - Home dir & project dir revisited: <lc_xx> group owner

HPCC accounts illustrated

- How should I call it, account or allocation?
 - Actually there are few things:
 - RCF unix account
 - Proper unix group affiliation: <hpcusers>, <lc_xx>
 - Qbank account
 - Naming convention
 - lc_ prefix for qbank account & unix group
 - qbank group account identical to unix group
 - qbank individual account identical to rcf username

More on HPCC accounts

- RCF account allow you to log in almaak
- <hpcusers> group allow you to log in hpc login nodes
- <lc_xx> groups allow your group members visit your project dir
- Qbank group account <lc_xx> to hold computing time
- Qbank individual account as member of group account allow you to compute using time in that group account

More on HPCC accounts

- Log in almaak, or hpcc login nodes, type
 - `getent passwd userid`
 - RCF user info will appear
 - `groups userid`
 - Your groups affiliation will appear
- Check the balance of qbank account
 - `qbalance -h -a lc_xx`

Examples on HPCC accounts

```
bash-2.05b$ qbalance -h -a lc_sn
```

```
Account:      lc_sn
```

```
Subaccount:   Available (Nodehours)
```

```
-----  
ANY           0.0000
```

```
KITTY        114102.9769
```

```
RESERVE      0.0000
```

```
bresch       0.0000
```

```
shrikann     0.0000
```

```
-----  
Total        114102.9769
```

```
bash-2.05b$ /usr/bin/groups pechpras
```

```
csci-ar hpcusers lc_pv1 lc_an2 lc_cmb
```

```
bash-2.05b$ getent passwd pechpras
```

```
pechpras:KERB:121365:5117:Thanakij Pechprasarn:/home/cmb-01/pechpras:/bin/tcsh
```

HPCC job queues

- What is a job?
 - typically a shell script with additional info
- What is a job queue?
 - A way to enforce group policy
 - Part of node property in perspective of scheduler, such as 'main', 'large'
 - Default queue: routing jobs

HPCC job queues cont'd

- HPCC queue properties
 - <http://www.usc.edu/hpcc/systems/newuser.php>

queue name	max jobs queued	max node count	max walltime	max jobs per user
main	1000	99	24 hr	10
quick	100	4	1 hour	10
long	100	64	336 hours	16
large	100	256	24 hours	N/A
largemem	N/A	1	336 hours	1

PBS system

- PBS: manages all resources, assigns resources to jobs according to Maui, controls job execution
- Maui: given all resources PBS presents, find solution for job requests and notify PBS

Job submission

- Command qsub
- Examples
- PBS script
 - How to write a script
- How to write efficient script: pbsdsh
- Examples

Command qsub

- Simplified syntax

```
qsub [-A account_string] [-I] [-l resource_list] [-q destination] [-X] [script]
```

- Most common parameters

- -A account_string: specifies the hpcc allocation where the wall time will be deducted
- -I : interactive job. Useful! Note: can't appear with [script]
- -l resource_list : list the resources that the job requests
- -q destination : specify the queue where the job goes to
- -X : enable X forwarding

More on qsub

- Default account
 - qjsuser –d userid
- Resource list
 - wall time: walltime=2:00:00
 - node count and ppn:
 - nodes=4
 - nodes=4:ppn=2
 - node type: arch=i686 or arch=x86_64

Examples of qsub

```
qsub myjob.pbs          # default queue, account, wall time, nodes, ppn
qsub -A lc_an myjob.pbs # with account <lc_an>
qsub -l walltime=00:45:00,nodes=2 myjob.pbs
qsub -q cmb myjob.pbs  # specify a condo queue
qsub -I                # interactive job
qsub -I -X             # interactive job with X forwarding enabled
qsub -l walltime=2:00:00,nodes=4:ppn=2,arch=x86_64 myjob.pbs
```

PBS script

- It's a set of commands that executes in batch to complete a specific user computing task
- How to write it?
 - Use interactive jobs `qsub -I`
 - Run the command one by one
 - When you are done, put all commands into one file
 - If you are running parallel program, use `mpiexec`, not `mpirun`

More on PBS script

- Make your qsub command more concise
 - Include resource request in the script
 - If you frequently use the same qsub options for a given script, PBS lets you put the options in the script itself so that you do not have to type them every time you submit a job.
 - Command line parameters overrides!
- Use predefined environment variable
 - `$PBS_O_WORKDIR`
 - Remembers the dir where you executed qsub

PBS script examples

- Make qsub more concise
 - `qsub -l walltime=2:00:00,nodes=4:ppn=2,arch=x86_64 myjob.pbs`
 - You would put the following lines at the beginning of myjob.pbs(after 1st line):
 - `#Run on 8 processors on uschpc`
 - `#PBS -l walltime=2:00:00`
 - `#PBS -l nodes=4:ppn=2`
 - `#PBS -l arch=x86_64`
 - `#PBS` line needs to be before any non-comment lines
 - Now you only need to type
 - `qsub myjob.pbs`

More on PBS script examples

- A complete script

```
#!/bin/bash
#PBS -l nodes=1:ppn=2
#PBS -l walltime=00:00:59
cd /home/rcf-proj3/pv/test/
source /usr/usc/sas/default/setup.sh
sas my.sas
```
- First line has to be the shell

Parallel Computing

- single processor job: not efficient
- Software can not run in parallel
- Solution: pbsdsh
 - Most common usage:

```
#!/bin/sh
#PBS -l nodes=4:ppn=2
#PBS -l walltime=5:00:00
... initial processing ....
pbsdsh myscript
... final processing ....
```

More on pbsdsh

- All tasks are identical
- If tasks are different...

- \$PBS_VNODENUM

- \$PBS_NODENUM

- Example:

```
#!/bin/bash
```

```
cd $PBS_O_WORKDIR
```

```
PATH=$PBS_O_PATH
```

```
mysub.$PBS_VNODENUM
```

Common mistakes

- Don't run jobs on hpc-login nodes
 - Shared resource
 - Always use qsub
- Script doesn't run as expected, shell error
 - Blank first line? Goes to default shell /bin/sh
- Do the editing on linux
 - Windows may insert some control characters

More on common mistakes

Look at this script please:

```
#!/bin/bash
#PBS -l nodes=20:ppn=8
#PBS -l walltime=09:59:59
cd /home/rcf-proj3/pv/test/
./myprog
```

Consider the following questions:

1. Will this run parallel?
2. If yes, how many processors does it use?

Job monitoring

- job id: 7-9 digit number that qsub returns, followed by the server name
- Useful commands
 - qstat
 - checkjob
 - showstart
 - pbstop

Command qstat

- used to request the status of jobs, queues, or a batch server
- Most often usage
 - `qstat` # list all jobs
 - `qstat -f jobid` # full status of the job
 - `qstat -u userid` # all jobs submitted by userid

checkjob and showstart

- How long will the job sit in the queue?
 - showstart jobid
- My job was queued for a while, why?
 - checkjob jobid
 - It will usually show you why:
 - Request of resources can not be met – be patient!
 - Something is wrong – what's the error message?

pbstop

- Draws a full-terminal display of your nodes and jobs.
- Use arrow keys to scroll up and down, space to refresh
- `/`: search nodename, job id
- `u`: prompt for a username to limit the view of the grid to the named user
- `l`: prompt for a job id. A node load report shows the current load average, the physical and available memory, and the number of sessions.
- `0-8`: toggle display of that CPU number in the display.
- `h`: help

More on job monitoring

- Output of a job
 - .o file
 - Prologue
 - Warning message
 - Warning: no access to tty (Bad file descriptor).
 - Thus no job control in this shell.
 - Standard output
 - Epilogue
 - .e file standard error
- Need more info(memory/CPU usage)?
 - Log in nodes when the job is still there

Common errors

- Job does not run
 - No enough balance in qbank account
 - Didn't specify the correct qbank account
 - Typos: x86_64 & X86_64...case sensitive

Tips of job control

- A job that can't be deleted:
 - PBS_MOM does not know the process exited
 - command
 - `qsig -s 0 jobid`
 - Will let mom look for the process and exit job
- Delete all jobs from a user:
 - command
 - `qselect -u userid | xargs qdel`

End

- Questions?